Playground Series Season 3, Episode 4.

Card Fraud Classification

Prepared By: Soumee Ghosh (2211444)

Data Description:

It is important that credit card companies are able to recognize fraudulent credit card transactions so that customers are not charged for items that they did not purchase.

The dataset contains transactions made by credit cards in September 2013 by European cardholders. This dataset presents transactions that occurred in two days, where we have 492 frauds out of 284,807 transactions. The dataset is highly unbalanced, the positive class (frauds) account for 0.172% of all transactions.

It contains only numerical input variables which are the result of a PCA transformation. Unfortunately, due to confidentiality issues, we cannot provide the original features and more background information about the data. Features V1, V2, … V28 are the principal components obtained with PCA, the only features which have not been transformed with PCA are 'Time' and 'Amount'. Feature 'Time' contains the seconds elapsed between each transaction and the first transaction in the dataset. The feature 'Amount' is the transaction Amount, this feature can be used for example-dependant cost-sensitive learning. Feature 'Class' is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Given the class imbalance ratio, we recommend measuring the accuracy using the Area Under the Precision-Recall Curve (AUPRC). Confusion matrix accuracy is not meaningful for unbalanced classification.

The analysis can be divided into the following sections:

1. Loading and understanding the data.

Firstly we load all the data wrangling, data visualization and machine learning libraries. We load the dataset using the pandas library into a dataframe. Thereafter we check the null values and datatypes of the columns in the dataframe. Using the .info() method we find that the training data consists of 32 features and 219129 observations. There are 30 features with float datatype and 2 features with integer datatype which consists of the “id” and “Class” feature which is the target feature.

Using the .isnull() method , we find that there is no null values present in the dataset.

We try to look for the distribution of the target column “Class” and we can see that it is highly imbalanced.

```
]: train_df.Class.value_counts()

]: 0    218660
   1       469
   Name: Class, dtype: int64
```

2. Data Cleaning and EDA

Thereafter we try to check the skewness of the columns.

```
Time     -1.003110e-01
V1       -1.665854e+00
V2       -2.173930e+00
V3       -1.288304e+00
V4        2.183217e-01
V5        5.478835e-01
V6        1.438684e+00
V7       -2.398198e-01
V8       -6.768427e+00
V9        4.479995e-01
V10       1.296309e+00
V11       6.665395e-02
V12      -1.161589e+00
V13       1.061609e-01
V14      -8.405754e-01
V15      -4.764631e-01
V16      -4.717247e-01
V17      -6.188927e-01
V18      -1.287733e-01
V19      -8.473591e-02
V20       1.778520e+00
V21       5.936353e+00
V22      -1.970778e-01
V23       3.181585e-01
V24      -3.737147e-01
V25      -4.529469e-01
V26       7.687305e-01
V27      -2.056500e+00
V28       1.283371e+01
Amount    9.047609e+00
```

So can observe that the following features are skewed:

skewed_columns=["V1","V2","V3","V6","V8","V10","V12","V20","V21","V27","V28","Amount"]

Using the interquantile range we try to remove the skewness.

Similarly we perform this for the test set.

Thereafter we check the correlation between the variables

3. Data Preprocessing & Model Building

We divide the training data into X and y variables where X contains all features except the target and “id” feature and y contains the target feature i.e. “Class”.

Thereafter we import the MinMaxScaler to fit and transform the features.

Finally we split the training data into train and validation set and perform stratified K fold classification through Decision Tree, Random Forest, logistic regression and XG boost classifier.

We compare the accuracy,precision, recall and f1 scores and find the best model to predict on the test set.

```
lean Accuracy:  0.9978597082915327
lean Precision:  0.2
lean Recall:  0.002127659574468085
lean F1 Score:  0.004210526315789474
------------------------------
lodel: Random Forest
lean Accuracy:  0.9978551447904145
lean Precision:  0.0
lean Recall:  0.0
lean F1 Score:  0.0
------------------------------
lodel: XGBoost
lean Accuracy:  0.9978505813934267
lean Precision:  0.5
lean Recall:  0.010661175932280942
lean F1 Score:  0.020793119836221393
------------------------------
)verall Results
lean Accuracy Scores:  [0.9978597082915327, 0.9978551447904145, 0.9978505813934267]
lean Precision Scores:  [0.2, 0.0, 0.5]
lean Recall Scores:  [0.002127659574468085, 0.0, 0.010661175932280942]
lean F1 Scores:  [0.004210526315789474, 0.0, 0.020793119836221393]
```

We find that XG boost is the best model for our prediction hence we predict the testing set using XGboost.